MCMC Methods for Entropy Optimization and Nonlinear Network Coding

Sormeh Shadbakht and Babak Hassibi
Electrical Engineering Department
California Institute of Technology
Pasadena, CA 91125
Email: sormeh,hassibi@caltech.edu

***Abstract*—Although determining the space of entropic vectors for n random variables, denoted by Γ_n^*, is crucial for solving a large class of network information theory problems, there has been scant progress in explicitly characterizing Γ_n^* for $n \geq 4$. In this paper, we present a certain characterization of quasi-uniform distributions that allows one to numerically stake out the entropic region via a random walk to any desired accuracy. When coupled with Monte Carlo Markov Chain (MCMC) methods, one may "bias" the random walk so as to maximize certain functions of the entropy vector. As an example, we look at maximizing the violation of the Ingleton inequality for four random variables and report a violation well in excess of what has been previously available in the literature. Inspired by the MCMC method, we also propose a framework for designing optimal nonlinear network codes via performing a random walk over certain truth tables. We show that the method can be decentralized and demonstrate its efficacy by applying it to the Vamos network and a certain storage problem from [1].**

I. Introduction

Let $X_1, X_2, \ldots, X_n$ to be n jointly distributed discrete random variables and for any nonempty set $\alpha \subseteq \{1, 2, \ldots, n\}$ define the joint entropy $h_\alpha = H(X_i : i \in \alpha)$. In conjunction, the joint entropies of all such nonempty subsets defines a vector in $\mathbb{R}^{2^n-1}$ called an entropy vector. The set of all entropy vectors derived from n random variables is denoted by Γ_n^* and its closure, $\bar{\Gamma}_n^*$ is well known to be a convex cone. The entropy region is of great importance since maximizing the (weighted) throughput for a large class of wired acyclic networks can be reduced to convex optimization over $\bar{\Gamma}_n^*$ [2], [3].

Despite the importance attached to the entropy region, there exists very little in the way of explicitly characterizing $\bar{\Gamma}_n^*$ for $n \geq 4$ random variables. Since the goal is most often to perform optimization over $\bar{\Gamma}_n^*$ (to solve a network information theory problem, say), in the absence of an explicit characterization of the entropy region, the next best thing is to present a method to *numerically* perform optimization over this region. Presenting such a numerical framework is the goal of the current paper.

The approach we shall take is via a characterization of all so-called *quasi-uniform* distributions. It is well known that this class of distributions is sufficient to approximate the entropic region to any fidelity. We present a random walk over this characterization of distributions which, when coupled with a suitable Monte Carlo Markov Chain (MCMC) method, allows for optimization of any function of the entropy vector. Furthermore, we show how the MCMC method can be used as a framework to design optimal nonlinear network codes in a distributed fashion via performing a random walk over certain truth tables. We demonstrate the efficacy of the method by looking at three examples: violating the Ingleton inequality, maximizing capacity of the Vamos network, and solving a certain storage problem from [1].

Of course, many issues remain unresolved, such as the convergence rate of the method, the choice of the temperature parameter, etc. The results presented here are preliminary, yet promising, and future work will delve into the issues much further.

II. Quasi-Uniform Distributions and Entropy Vectors

At first sight, the difficulty in characterizing $\bar{\Gamma}_n^*$ appears to be that one must consider *all* possible joint distributions of n random variables for *all* alphabet sizes. However, it turns out that there is a certain class of distributions that suffices.

A distribution is called *quasi-uniform* [4] if its probability mass function, as well as the probability mass function of all its marginals, takes on a constant or zero value on all points in the sample space. An example of a quasi-uniform distribution in two variables is given in Fig. 1, where each "x" means that a constant nonzero probability of $\frac{1}{24}$ is assigned to that point in the sample space. As can be seen, one marginal is uniform with probability $\frac{1}{8}$ and the other is quasi-uniform with probabilities 0 and $\frac{1}{6}$.

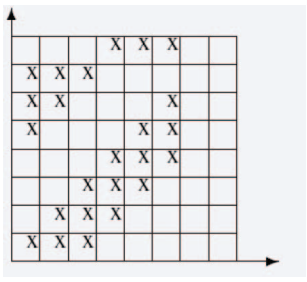

Fig. 1. An example of a quasi-uniform distribution

Let Λ_n denote the space of entropy vectors generated by quasi-uniform distributions.

Theorem 1 (Quasi-Uniform Distributions)**.** *[4] The closure of the cone of Λ_n is the closure of Γ_n^*.*

Thus, as promised earlier, quasi-uniform distributions suffice to characterize the entropy region. Though at first sight it may seem surprising, however, this result can be motivated by recourse to the concept of "typical sequences". To this end, make T independent copies of each of our random variables, to get vector-valued sequences of length T. As $T \to \infty$, with probability approaching one, we will only encounter typical sequences. If we assign a constant probability to all typical sequences, and zero probability to nontypical ones, it is straightforward to see that we end up with a quasi-uniform distribution with the same entropy vector. The entropy is simply the log of the number of typical sequences divided by T, which for a random variable of alphabet size N is

$$h \simeq \frac{1}{T} \log \frac{T!}{T_1!T_2! \dots T_N!}, \qquad T_i = Tp_i, \qquad \sum_{i=1}^{N} T_i = T. \quad (1)$$

In fact, this is essentially the statistical physics interpretation of entropy. However, (1) can also be interpreted in terms of subgroups of the permutation group on T elements. $T!$ is simply the size of the permutation group, whereas if we partition the T elements into disjoint sets of size T_i each, then $T_1!T_2! \dots T_N!$ is simply the size of the subgroup of permutations that respects this partition. This leads to a connection between entropy and groups. However, we shall not further delve into this here.

Although, based on Theorem 1, determining all the quasi-uniform distributions is equivalent to characterizing Γ_n^*, it appears that determining all quasi-uniforms is a hard combinatorial problem. In the next section, we shall use (1) to characterize all possible entropy vectors of quasi-uniform distributions and to propose a random walk over them.

III. Entropy Optimization

A. A Characterization of Quasi-Uniform Distributions

As mentioned in the previous section, determining all quasi-uniform distributions seems to be a hard combinatorial problem. An idea to tackle this problem is to try to do a random walk on such distributions. However in order to do so, we need a method that 1) determines how to move from any quasi-uniform to any other such distribution, therefore defining an irreducible Markov chain and 2) exhausts all quasi-uniforms. Working with distribution tables as the one in Figure 1, quickly reveals that, devising a method to move from one quasi-uniform distribution to another is highly nontrivial. On the other hand, given that any entropy can be approximated by (1), one can chacterize the entropies of quasi-uniform distributions (by characterizing all possible partitions and joint partitions of T elements) and then perform a random walk on the entropy vectors. The idea is as follows.

Let n be the number of random variables. Choose values T and N and construct a $T \times n$ table with entries drawn from the set $\{0, 1, \dots, N-1\}$. Each column of the table corresponds to one of the random variables and induces a partition of T elements into at most N disjoint sets, if we let the entries with the same value belong to the same partition. The entropy of the corresponding random variable is simply computed from (1) using this induced partition. Similarly, any q-tuple of columns defines a partition of the T elements into at most N^q disjoint sets (identical rows belong to the same partition). Again, the joint entropy of the corresponding collection of random variables is computed from (1) using this induced partition [5].

Example: Consider the following table of size $T = 5$ by $n = 2$ with $N = 2$:

$$\begin{bmatrix} 0 & 1 \\ 1 & 0 \\ 1 & 0 \\ 1 & 1 \\ 0 & 1 \end{bmatrix}$$

The partitions for the first column will be $T_1 = |\{0,0\}| = 2$ and $T_2 = |\{1,1,1\}| = 3$ whose corresponding quasi-uniform entropy will be $h_1 = \log_2 \frac{5!}{2!3!} = \log_2 10$. For the second column the partitions are similarly $T_1 = |\{0,0\}| = 2$ and $T_2 = |\{1,1,1\}| = 3$ giving $h_2 = \log_2 \frac{5!}{2!3!} = \log_2 10$, and finally for both columns the partitions are, $T_1 = |\{(0,1),(0,1)\}| = 2$, $T_2 = |\{(1,0),(1,0)\}| = 2$ and $T_3 = |\{(1,1)\}| = 1$ resulting in $h_{12} = \log_2 \frac{5!}{2!2!1!} = \log_2 30$.

Lemma 1. *Every such $T \times n$ table corresponds to a quasi-uniform distribution. Furthermore, as T and N grow, we encounter the set of all quasi-uniform distributions over n variables that are sufficient to characterize Γ_n^*.*

Proof: Assume that the given table corresponds to T independent copies of n random variables with alphabet size N. Then from the above definition of partition on T elements and also (1), it is clear that we can assign a permutation group G on T elements and define its subgroups G_i as the ones that permute within each partition. It is then straightforward to generate quasi-uniform distributions from groups. Let the alphabet size of each random variable be $\frac{|G|}{|G_i|}$, i.e., the number of cosets induced by each G_i. For each element $g \in G$, obtain a n-dimensional vector whose i-th component is the index of the coset induced by G_i that it belongs to. Assign a constant probability to every vector in the sample space encountered in this fashion, and assign zero probability to all other vectors in the sample space. It is not too difficult to see that the resulting distribution is quasi-uniform whose entropy is obtained from, $\log \frac{T!}{T_1!T_2!\dots T_N!}$. Therefore to every $T \times n$ table, we can assign a quasi-uniform distribution. Moreover as N and T grow, we allow for all alphabet sizes of distributions and also make the approximation (1) more precise which means that we will asymptotically encounter the set of all the quasi-uniform distributions over n random variables that characterize Γ_n^*. ■

Remark: Note that for fixed N and T, the space of such obtained quasi-uniform distributions is connected. In other words one can move from a quasi-uniform distribution, corresponding to a table A, to another quasi-uniform distribution, corresponding to a table B, by a chain of changes in the entries of the table that transforms table A to table B. We can thus perform a random walk over quasi-uniform distributions by

randomly choosing an entry of the $T \times n$ table and randomly changing its value. In this manner we can numerically stake out the entropic region.

Of course, to numerically stake out the entropy region with higher and higher fidelity requires one to increase the values of T and N. This results in an increase in the size of the search space and slows down the MCMC methods we describe next. Thus, there is a trade-off between the quality of the results and the speed of the optimization program. Choosing the right T and N may therefore be of critical importance.

B. Entropy Optimization via Gibbs Sampling

Assume that we have the following optimization problem,

$$\max_{h \in \Gamma_n^*} f(h), \tag{2}$$

where $f(\cdot)$ is some function of the entropy vector. Following the arguments of the last section, we can define a random walk on the quasi-uniform distribution of n random variables for fixed T and N and use Gibbs sampling to perform the optimization. To do so, we first generate a $T \times n$ table either randomly, or by initializing it to some desired value. Let the corresponding cost of this table be denoted by $c = f(h)$. Next, at each step we choose an entry of the table at random and change its value to any other of the $N-1$ possible choices randomly. Let the new cost be $c' = f(h')$. We decide to accept the changed entry, i.e. the new table, with probability,

$$p_{cc'} = \frac{e^{\theta c'}}{e^{\theta c'} + e^{\theta c}} \tag{3}$$

where θ is a parameter usually called the temperature in the Gibbs sampling (a.k.a the heat bath method). Note that if we assume that each quasi-uniform represents a state, we have defined a Markov chain on the state-space of quasi-uniforms with a transition probability (3). Moreover note that this Markov chain is irreducible simply because there is a path between any two states. Let,

$$\pi_{c'} = \frac{e^{\theta c'}}{\sum_{c_i} e^{\theta c_i}} \tag{4}$$

where c_i denotes the cost of a feasible table. Then we have, $\pi_c p_{cc'} = \pi_{c'} p_{c'c}$, meaning that π is a reversible distribution and hence a stationary distribution for the Markov chain. Therefore if we do a random walk on this Markov chain with transition probability (3) for a long time, the Markov chain converges to the distribution (4). Note that, when θ is large, the stationary distribution will have a large peak at the optimal cost and therefore the chances of encountering it (once we are in steady state) is high. However, a large θ often means that convergence to the steady state distribution can be slow (we may frequently get stuck in local maxima) as (3) heavily favors transitions to higher costs. On the other hand, for small values of θ, convergence to the steady state is much faster (as (3) is more amenable to escape from local maxima). However, the peak in (4) is not very pronounced at the optimal cost and so it might take a very long time until we encounter it.

Therefore there is a trade-off between speed of convergence to the stationary distribution and the probability of encountering the optimal cost, once the Markov chain has converged. And so choice of the correct value of θ is critical and may require trial and error.

C. Ingleton Violation via MCMC

As an application of the Gibbs sampler just described, we consider maximizing the violation of the Ingleton inequality. The Ingleton inequality holds for entropy vectors obtained from linear network codes (and more generally from Abelian groups) and is given by

$$\Delta_{ij} \triangleq h_i + h_j + h_{ijk} + h_{ijl} + h_{kl} - h_{ij} - h_{ik} - h_{il} - h_{jk} - h_{jl} \leq 0. \tag{5}$$

However, the Ingleton inequality is not a bound on the entropy region and there exist entropy vectors that violate it. We define the "violation index" as $\frac{\Delta_{ij}}{\|h\|}$. Note that the normalization is critical as entropy is a cone. Furthermore, the violation index is proportional to the cosine of the angle between the vector h and the vector orthogonal to the Ingleton plane.

Using the MCMC method with values $T = 100$ and $N = 2$ and $\theta = 50000$, we found a violation index of 0.0249 which is more than triple 0.0073, the best violation of Ingleton available in the literature (a certain distribution based on projective planes given in [6]). It is also far better than the maximum violation index value of Ingleton violating example $PGL(2, p)$ of [7] whose maximum value 0.0082 occurs for $p = 13$. See Fig. 2 for a sample run of the MCMC method.

IV. Nonlinear Network Coding

The above idea of a random walk over quasi-uniform distributions can be extended to a random walk over all (possibly nonlinear) operations in a network.

Assume that vector-valued signals of size l over alphabet size N are transmitted across edges of the network [1]. In such a setup, if the in-degree of a particular node in the network is D, then the node must map each of its possible $N^{l \times D}$ inputs to its corresponding outputs. For each output, this mapping can be represented by a truth table with $D + 1$

[1] In general, source variables and middle variables of the network can be of different vector sizes.

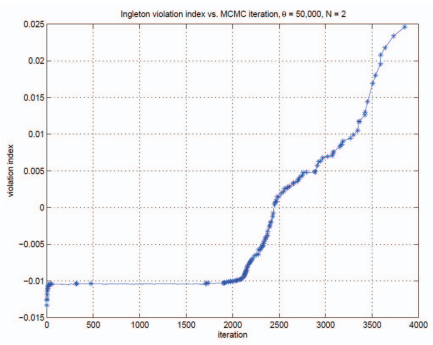

Fig. 2. A sample run of MCMC for Ingleton violation

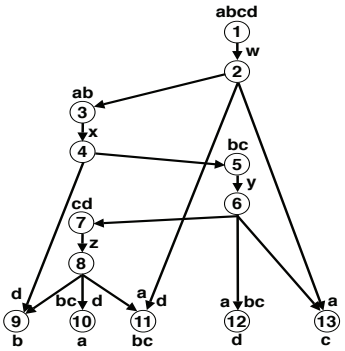

Fig. 3. The Vamos network

block columns of size l, the last block column representing the output, and $N^{l\times D}$ rows (one for each input combination). There are a total of $N^{l\times N^{l\times D}}$ possible truth tables, and thus a total of $N^{l\times N^{l\times D}}$ possible nonlinear network operations, for this particular output of the internal node. On the other hand, note that, if we restrict ourselves to linear mappings, there will only be $N^{l^2\times D}$ possible mappings for this node. The total number of network operations is obtained from the conjunction of the possible operations for each internal node and can be computed to be:

$$N^{\sum_{j\in\nu, j\notin\mathcal{S}_{source}\cup\mathcal{S}_{sink}} l\times|Out(j)|N^{l\times|In(j)|}}, \qquad (6)$$

where $|Out(j)|$ and $|In(j)|$ are the out- and in-degree of the node j.

A random walk over these network operations can be performed by simply randomly choosing an internal node, randomly choosing an output of this node, randomly choosing a row of its truth table, and then randomly changing the output value corresponding to this input row. If we define as the cost function the weighted sum rate, then a "heat bath" idea, can be employed to bias the random walk to large costs, and thereby to network operations that yield a large weighted sum rate.

A. Random Walk on Truth Tables

We have applied this method to two networks of interest and in what follows we give the simulation results and their comparison with the existing results.

1) Vamos Network: The Vamos network (Fig. 3) is obtained from the well-known Vamos matroid and was first introduced in [8] where the authors showed that the network is not solvable and proved the insufficiency of Shannon-type information inequalities for determining the capacity of general networks, reaffirming the importance of the full characterization of Γ_n^*. However using a non-Shannon type information inequality they provided an upperbound of 10/11 for the network coding capacity. They also found the linear coding capacity of the Vamos network to be 5/6 over every finite field and gave a vector-valued solution of size 6. In this network a, b, c, d are sources and x, y, z, w are internal messages. The sinks are nodes 9 to 13 whose demands are shown below them.

We employ the heat bath method for this network and as stated in the previous section do a random walk on its truth tables. Since there are 4 message variables in the network, there will be a total of 4 truth tables, for x, y, z and w respectively. Assuming the simplest case, we considered $N = 2$, i.e. binary alphabet size and scalar valued random variables for all the source and message variables and performed the following maximization of normalized sum rate,

$$\max \frac{1}{6}(I(b;dz) + I(a;bcdz) + I(bc;adzw) + I(d;abcy) + I(c;awy)) \qquad (7)$$

As can be seen from Fig. 4, the simulation gets to the linear capacity of 5/6 = 0.8333 in only a few hundred iterations. Although there is the 10/11 upperbound for this network, running the simulation for about 6000 iterations, did not result in a better sum rate than 5/6, raising the possibility that this is the best achievable rate among all scalar binary codes.

Remark: If the optimization (7) results in a normalized sum rate less than 1, then one or more sinks do not fully reconstruct their demands (e.g. if $I(b;dz^*) < 1$, where $(.)^*$ denotes using the MCMC solution to (7), then b is not fully recovered in sink 9). Since the sources are assumed to be i.i.d uniform, a random coding argument along with random binning may be used to deliver to each sink its corresponding rate as calculated in (7) (e.g. to deliver rate $I(b;dz^*)$ to sink 9). Clearly if (7) results in an optimum value 1, truth tables of MCMC provide a valid solution for the network.

2) Repair Problem in a Storage System: In distributed storage systems, one usually encounters the repair problem, i.e., if a node of the system which is storing a piece of encoded data fails or leaves the network, to maintain the same level of reliability, other surviving nodes should be able to recover the lost encoded data with access only to the other encoded data (of the non-source nodes) [1]. Here we consider a special case of the storage problem, in which there are 2 sources and 4 storage nodes. The goal is to find codes that enable us to recover the original data from any two storage points and also be able to recover the exact encoded data of any storage node from the remaining storage nodes once that node fails or leaves the system. Fig. 5 shows the structure of this network.

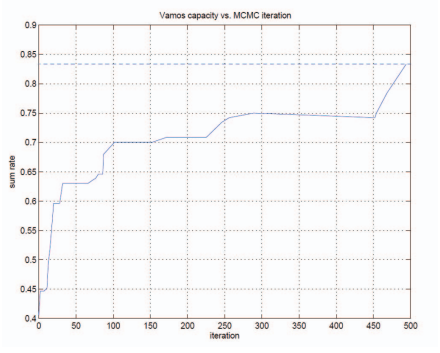

Fig. 4. The Vamos capacity

[1] refers to this problem as the "exact repair" problem. By sticking to linear "rotationally symmetric codes" the authors reduce the space of potential codes and do an exhaustive search over this space to find regenerating codes. We have taken this network and employed the Monte Carlo method over GF(3) for which [1] reports finding rotationally symmetric codes. Again we have considered maximizing the following sum rate,

$$\sum_{i,j\in\{1,2,3,4\}} I(X_{1s}X_{2s}; X_iX_j) + \sum_{\substack{i,\,j,\,k,\,l\,\in\,\{1,\,2,\,3,\,4\}\\ i\,\in\,\{1,\,2,\,3,\,4\}\,\setminus\,\{j,\,k,\,l\}}} I(X_i; X_{ji}X_{ki}X_{li}) \quad (8)$$

where X_{is} and X_i are the source signal and the ith encoded stored signal respectively and X_{ji} denotes the outgoing signal from the storage node of X_j that is used for recovery of the lost node say, X_i. Moreover, we have considered vector sizes of 2 for source variables X_{is} and also the encoded data X_i and scalar values for X_{ji}. Note that the maximum possible value for the objective function is equal to 50.7188 and is obtained when all the destinations can recover their intended signals and h_{X_i} are at their maximum, in this case equal to $2\log_2 3$.

We have studied the network over linear operations. Simulation results show (Fig. 6), that the MCMC method finds the solution in only a few hundred iterations. We have seen that Monte Carlo methods can be used for entropy optimizations or in networks to find the best sum rate under certain conditions. In practice, especially for large networks, one would want to employ such methods in a distributed manner. In fact, this is easily done, as described next.

B. Distributed MCMC over Networks

Algorithm 1 (Distributed Training "Proto"-Algorithm)**.** *The network operations at each internal are initially set to some fixed or random operations. Assume that there are r independent sources, the random variables of the network are of size l and operations are over GF(N). The algorithm consists of T training epochs. During each training epoch:*

1) *Each source transmits a packet of length $N^{l\times r}$, representing one column of a r-input truth table. In conjunction, these $N^{l\times r}$ channel uses per training epoch represent all possible inputs to the network.*
2) *One (or more) internal nodes randomly choose themselves (a la Aloha). A chosen internal node performs a random step on its local truth table (as explained before)*

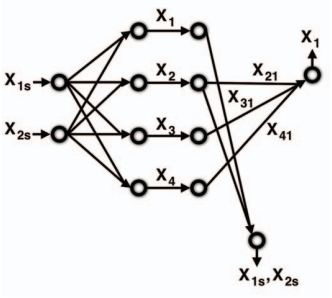

Fig. 5. Storage problem

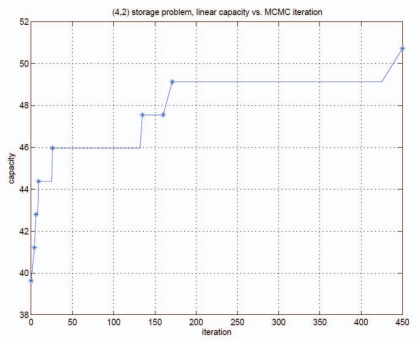

Fig. 6. Storage problem capacity

and implements the new truth table on the input signals it sees during the training epoch.

3) *At the end of the training epoch, the sink nodes can compute their recovery rates because they can compute the mutual information between their received signals and their desired inputs (because they know the transmitted inputs and have measured the corresponding outputs).*
4) *These recovery rates are fed back to the network so that every node can compute the new weighted sum rate.*
5) *The chosen internal node(s) compare the new weighted sum rate with the old one and randomly choose to keep their new truth table according to a "heat bath" or "simulated annealing" step.*

During the training process all nodes store the largest weighted sum rate encountered and their respective truth tables corresponding to it. At the end of the T training epochs, the internal nodes set their truth tables to these best encountered ones. Data transmission at the best rates found can now commence.

When the number of sources in the network is not too large, say $r \leq 8$, and the alphabet size is binary, the packet lengths are not too long and it is conceivable that many thousand training epochs can be performed with ease. This will lead to distributed discovery of a network operation with high weighted sum rate.

REFERENCES

[1] D. Cullina, A. Dimakis, and T. Ho, "Searching for minimum storage regenerating codes," *arXiv:0910.2245v*.
[2] X. Yan, R. W. Yeung, and Z. Zhang, "The capacity region for multi-source multi-sink network coding," in *IEEE Int'l. Symp. Inf. Theory (ISIT)*, 2007.
[3] B. Hassibi and S. Shadbakht, "Normalized entropy vectors, network information theory and convex optimization," in *Inf. Theory Workshop, Bergen, Norway*, 2007.
[4] T. H. Chan, "A combinatorial approach to information inequalities," *Comm. in Inf. and Sys.*, vol. 1, no. 3, pp. 241–254, 2001.
[5] T. H. Chan and R. W. Yeung, "On a relation between information inequalities and group theory," *IEEE Tran. on Inf. Theory*, vol. 48, no. 7, pp. 1992–1995, 2002.
[6] Z. Zhang and R. Yeung, "On characterization of entropy function via information inequalities," *IEEE Trans. on Inf. Theory*, vol. 44, no. 4, pp. 1440–1452, 1998.
[7] W. Mao and B. Hassibi, "Violating the Ingleton inequality with finite groups," *46th Allerton Conf. on Comm., Control, and Comp.*, 2009.
[8] R. Dougherty, C. Freiling, and K. Zeger, "The Vamos network," *NETCOD*, 2006.